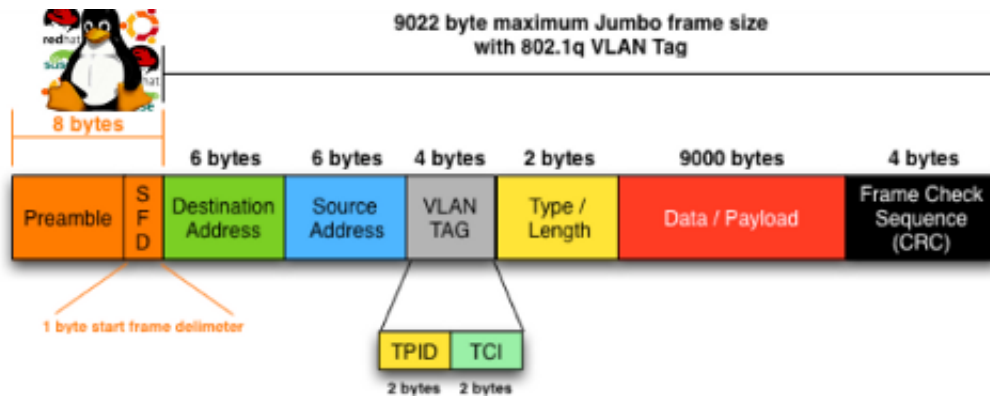


## Boost local network performance (Increase network throughput) by enabling Jumbo Frames on GNU / Linux

Author : admin



So **what is Jumbo Frames?** and why, when and how it can increase the network throughput on Linux?

Jumbo Frames are Ethernet frames with more than 1500 bytes of payload. They can carry up to 9000 bytes of payload. Many Gigabit switches and network cards supports them.

Jumbo frames is a networking standard for many educational networks like *AARNET*. Unfortunately most commercial ISPs doesn't support them and therefore enabling Jumbo frames will rarely increase bandwidth throughput for information transfers over the internet.

Hopefully in the years to come with the constant increase of bandwidths and betterment of connectivity, *jumbo frames* package transfers will be supported by most ISPs as well.

*Jumbo frames* network support is just great for is small local - home networks and company / corporation office intranets.

Thus enabling *Jumbo Frame* is absolutely essential for "local" ethernet networks, where large file transfers occur frequently. Such networks are networks where, there is often a Video or Audio streaming with high quality like **HD quality** on servers running File Sharing services like **Samba**, local **FTP** sites, **Webservers** etc.

One other advantage of enabling *jumbo frames* is **reduce of general server overhead** and decrease in CPU load / (*CPU usage*), when transferring large or enormous sized files. Therefore having **jumbo frames enabled on office network routers** with **GNU / Linux** or any other **\*nix** OS is vital.

*Jumbo Frames* traffic is supported in GNU / Linux kernel since version 2.6.17+ in earlier 2.4.x it was possible through external third party kernel patches.

### 1. Manually increase MTU to 9000 with ifconfig to enable Jumbo frames

```
debian:~# /sbin/ifconfig eth0 mtu 9000
```

The default MTU on most GNU / Linux (if not all) is **1500**, to check the default set MTU with *ifconfig*:

```
linux:~# /sbin/ifconfig eth0|grep -i mtu
UP BROADCAST MULTICAST MTU:1500 Metric:1
```

To take advantage of **Jumbo Frames**, all that has to be done is *increase the default Maximum Transmission Unit from 1500 to 9000*

For those who don't know **MTU is the largest physical packet size** that can be transferred over the network. MTU is measured by default in bytes. If a information has to be transferred over the network which exceeds the lets say 1500 MTU (bytes), it will be chopped and transferred in few packs each of 1500 size.

**MTUs** differ on different network topologies. Just for info here are the few main MTUs for main network types existing today:

- 16 MBit/Sec Token Ring - default MTU (17914)
- 4 Mbits/Sec Token Ring - default MTU (4464)
- FDDI - default MTU (4352)
- Ethernet - def MTU (1500)
- IEEE 802.3/802.2 standard - def MTU (1492)
- X.25 (dial up etc.) - def MTU (576)
- Jumbo Frames - def max MTU (9000)

Setting the *MTU packet frames to 9000* to enable **Jumbo Frames** is done with:

```
linux:~# /sbin/ifconfig eth0 mtu 9000
```

If the command returns nothing, this most likely means now the server can communicate on *eth0* with MTUs of each 9000 and therefore the network throughput will be better. In other case, if the network card driver or card is not a gigabit one the cmd will return error:

**SIOCSIFMTU: Invalid argument**

## 2. Enabling Jumbo Frames on Debian / Ubuntu etc. "the Debian way"

a.) *Jumbo Frames on ethernet interfaces with static IP address assigned* Edit **/etc/network/interfaces** and you should have for each of the interfaces you would like to set the *Jumbo Frames*, records similar to:

Raising the MTU to 9000 if for one time can be done again manually with *ifconfig*

```
debian:~# /sbin/ifconfig eth0 mtu 9000
```

```
iface eth0 inet static
address 192.168.0.5
network 192.168.0.0
gateway 192.168.0.254
netmask 255.255.255.0
mtu 9000
```

For each of the interfaces (eth1, eth2 etc.), add a chunk similar to one above changing the changing the IPs, Gateway and Netmask.

If the server is with two gigabit cards (eth0, eth1) supporting Jumbo frames add to */etc/network/interfaces* :

```
iface eth0 inet static
address 192.168.0.5
network 192.168.0.0
gateway 192.168.0.254
netmask 255.255.255.0
mtu 9000
```

```
iface eth1 inet static
address 192.168.0.6
network 192.168.0.0
gateway 192.168.0.254
netmask 255.255.255.0
mtu 9000
```

*b.) Jumbo Frames on ethernet interfaces with dynamic IP obtained via DHCP*

Again in */etc/network/interfaces* put:

```
auto eth0
iface eth0 inet dhcp
post-up /sbin/ifconfig eth0 mtu 9000
```

### 3. Setting Jumbo Frames on Fedora / CentOS / RHEL "the Redhat way"

Enabling jumbo frames on all Gigabit lan interfaces (eth0, eth1, eth2 ...) in *Fedora / CentOS / RHEL* is done through files:

- */etc/sysconfig/network-script/ifcfg-eth0*

- `/etc/sysconfig/network-script/ifcfg-eth1`

etc. ...

append in each one at the end of the respective config:

**MTU=9000**

```
[root@fedora ~]# echo 'MTU=9000' >> /etc/sysconfig/network-scripts/ifcfg-eth
```

a quick way to set **Maximum Transmission Unit** to 9000 for all network interfaces on Redhat based distros is by executing the following loop:

```
[root@centos ~]# for i in $(echo /etc/sysconfig/network-scripts/ifcfg-eth*); do \echo 'MTU=9000' >> $i
done
```

P.S.: Be sure that all your interfaces are supporting *MTU=9000*, otherwise increase while the MTU setting is set will return **SIOCSIFMTU: Invalid argument** err.

The above loop is to be used only, in case you have a group of identical machines with Lan Cards supporting Gigabit networks and loaded kernel drivers supporting MTU up to 9000.

Some Intel and Realtek Gigabit cards supports only a maximum MTU of 7000, 7500 etc., so if you own a card like this check what is the max MTU the card supports and set it in the lan device configuration. If increasing the MTU is done on remote server through SSH connection, be extremely cautious as restarting the network might leave your server inaccessible.

To check if each of the server interfaces are "Gigabit ready":

```
[root@centos ~]# /sbin/ethtool eth0|grep -i 1000BaseT
1000baseT/Half 1000baseT/Full
1000baseT/Half 1000baseT/Full
```

If you're 100% sure there will be no troubles with enabling MTU > 1500, initiate a network reload:

```
[root@centos ~]# /etc/init.d/network restart
...
```

#### 4. Enable Jumbo Frames on Slackware Linux

To list the ethernet devices and check they are Gigabit ones issue:

```
bash-4.1# lspci | grep [Ee]ther
```

0c:00.0 Ethernet controller: D-Link System Inc Gigabit Ethernet Adapter (rev 11)

0c:01.0 Ethernet controller: D-Link System Inc Gigabit Ethernet Adapter (rev 11)

Setting up *jumbo frames* on Slackware Linux has two ways; the slackware way and the "universal" Linux way:

a.) the Slackware way

On Slackware Linux, all kind of network configurations are done in **/etc/rc.d/rc.inet1.conf**

Usual config for eth0 and eth1 interfaces looks like so:

```
# Config information for eth0:
IPADDR[0]="10.10.0.1"
NETMASK[0]="255.255.255.0"
USE_DHCP[0]="""
DHCP_HOSTNAME[1]="""
# Config information for eth1:
IPADDR[1]="10.1.1.1"
NETMASK[1]="255.255.255.0"
USE_DHCP[1]="""
DHCP_HOSTNAME[1]="""
```

To raise the MTU to 9000, the variables **MTU[0]="9000"** and **MTU[1]="9000"** has to be included after each interface config block, e.g.:

```
# Config information for eth0:
IPADDR[0]="172.16.1.1"
NETMASK[0]="255.255.255.0"
USE_DHCP[0]="""
DHCP_HOSTNAME[1]="""
MTU[0]="9000"
# Config information for eth1:
IPADDR[1]="10.1.1.1"
NETMASK[1]="255.255.255.0"
USE_DHCP[1]="""
DHCP_HOSTNAME[1]="""
MTU[1]="9000"
```

```
bash-4.1# /etc/rc.d/rc.inet1 restart
```

```
...
```

b.) The "Universal" Linux way

This way is working on most if not all Linux distributions.

Insert in **/etc/rc.local**:

```
/sbin/ifconfig eth0 mtu 9000 up  
/sbin/ifconfig eth1 mtu 9000 up
```

### 5. Check if Jumbo Frames are properly enabled

There are at least two ways to display the **MTU** settings for eths.

*a.) Using grepping the MTU from **ifconfig***

```
linux:~# /sbin/ifconfig eth0|grep -i mtu  
UP BROADCAST MULTICAST MTU:9000 Metric:1  
linux:~# /sbin/ifconfig eth1|grep -i mtu  
UP BROADCAST MULTICAST MTU:9000 Metric:1
```

*b.) Using **ip** command from **iproute2** package to get MTU*

```
linux:~# ip route get 192.168.2.134  
local 192.168.2.134 dev lo src 192.168.2.134  
cache mtu 9000 advmss 1460 hoplimit 64
```

```
linux:~# ip route show dev wlan0  
192.168.2.0/24 proto kernel scope link src 192.168.2.134  
default via 192.168.2.1
```

You see MTU is now set to 9000, so the two server lans, are now able to communicate with increased network throughput.

Enjoy the **accelerated network transfers** ;)